# Demonstrating LLM-for-X: Application-agnostic Integration of Large Language Models to Support Writing Workflows

Lukas Teufelberger
Xintong Liu
Zhipeng Li
Department of Computer Science
ETH Zürich, Switzerland
Lukas.Teufelberger@inf.ethz.ch
Xintong.Liu@inf.ethz.ch
Zhipeng.Li@inf.ethz.ch

Max Moebus
Christian Holz
Department of Computer Science
ETH Zürich, Switzerland
Max.Moebus@inf.ethz.ch
Christian.Holz@inf.ethz.ch

## Abstract

In this demonstration, we show *LLM-for-X*, a system-wide shortcut layer that connects any application to backend LLM support through a lightweight popup dialog. LLM-for-X provides users with quick and easy-to-use LLM assistance without context switching to support writing and reading tasks. We show the use of LLM-for-X across several applications, such as Microsoft Office, VSCode, and Adobe Acrobat, which our tool seamlessly connects to the backends of OpenAI ChatGPT and Google Gemini. We also demonstrate the use of our system inside web apps such as Overleaf.

## CCS Concepts

• **Human-centered computing** → *Text input*.

## Keywords

Document authoring, Productivity tasks, Large Language Models.

**ACM Reference Format:**
Lukas Teufelberger, Xintong Liu, Zhipeng Li, Max Moebus, and Christian Holz. 2024. Demonstrating LLM-for-X: Application-agnostic Integration of Large Language Models to Support Writing Workflows. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST Adjunct '24), October 13–16, 2024, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3672539.3686757

## 1 Introduction

With the widespread availability of various Large language model (LLM) services, many users are now integrating them into authoring processes such as writing, editing, and question answering. Interaction with LLMs is mainly performed through chat interfaces and question & answer dialogs. Users can retrieve information, conduct research, and solve their problems [6, 10]. This has made chat LLMs a promising alternative to conventional search interfaces [9].

LLMs can be the key for next-generation intelligent personal assistants to operate as omnipresent personal companions. With the prolific progress in open-source NLP research, there are many studies on specialized bots and apps for specific use cases such as domain-specific or task-orientated text generation [8, 11]. Popular uses for LLMs include writing [1, 14] and coding assistance [3].

Beyond questions & answers, users frequently copy and paste content into and out of these chat interfaces to effectively execute a variety of tasks inside *other* applications. This workflow uses the clipboard as the interface to transfer textual information back and forth across tabs, windows, and applications. For many users, copy & paste is thus the dominant interface to bridge LLM assistance and the apps they use to complete tasks in a desktop environment.

For the design of companion systems, developers recommend that underlying tasks be disrupted as little as possible [15], as these can otherwise significantly impair productivity and quality of work [2, 4]. Even interruptions as small as context switching already affect productivity [7] but also one's mood [5].

In this demonstration, we show our interaction prototype *LLM-for-X*, a lightweight UI dialog that provides interaction with LLM backends for any frontend app. LLM-for-X requires no copy & paste or switching windows to LLM frontends, and it allows users to almost directly operate on text inside an app. Our system service connects text selections and user queries to an LLM by either emulating user input to a chat interface (e.g., an existing subscription for ChatGPT) and seamlessly transfers responses back into the app. Alternatively, LLM-for-X connects to LLM backend APIs to retrieve responses. In both cases, our approach minimizes the number of needed steps, allowing users to focus on a task within an app without context switching. See the full manuscripts for details [12].

## 2 Interacting through LLM-for-X

Figure 1 shows a series of interaction scenarios that are facilitated by LLM-for-X. Users summon LLM-for-X's dialog via `Alt` + `1` as a shortcut and type in an LLM query for quick execution. If they had selected text inside an app beforehand, this is used as the basis for the LLM query. Our dialog presents the LLM response in a preview field with before and after highlighting as useful. Users can also invoke shortcuts to execute frequently used operations instead. Finally, pressing `TAB` lets the LLM response flow back into the app from within LLM-for-X had been triggered. While our interaction flow is optimized for quick access and keyboard use, all options are also available via mouse use.

The visual design if LLM-for-X's popup is lightweight to minimize distraction from the foreground app. LLM-for-X's UI has three main elements: query input field (which is in focus when the menu appears), suggested actions (chosen based on popular use-cases of ChatGPT [13], accessible via shortcuts), and the preview output


*UIST Adjunct '24, October 13–16, 2024, Pittsburgh, PA, USA*

ACM ISBN 979-8-4007-0718-6/24/10
https://doi.org/10.1145/3672539.3686757

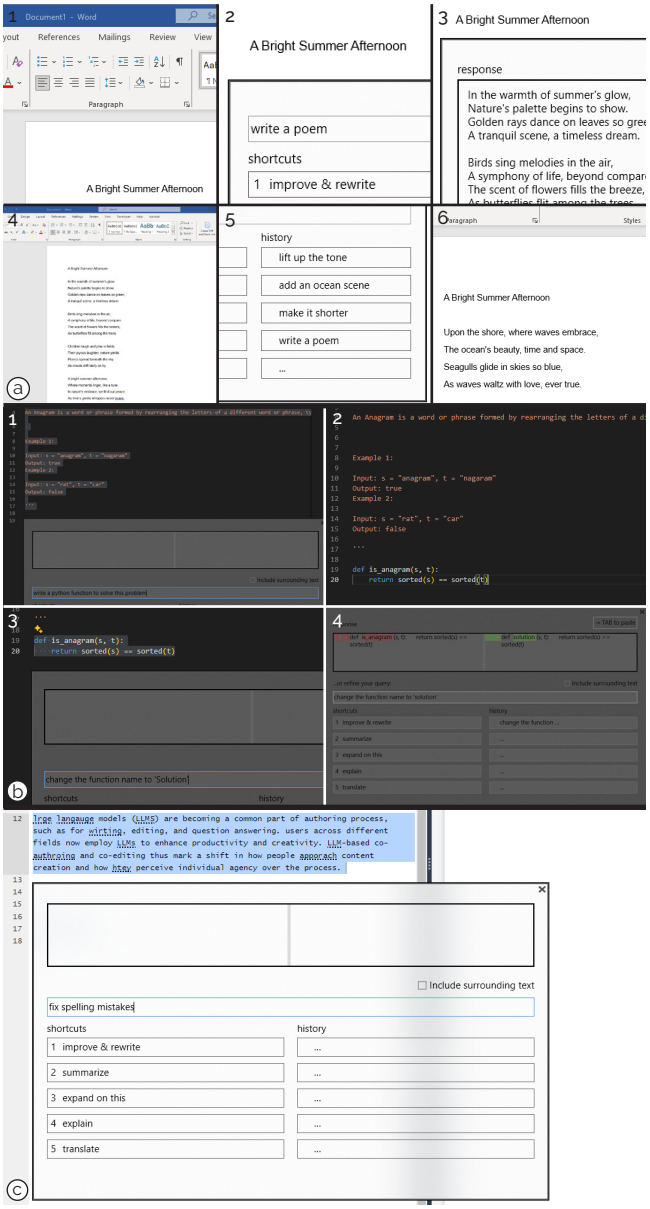


**Figure 1: LLM-for-X walk-through. (a) iterating on LLM responses, (b) coding with diff view, (c) full menu layout in context of writing assistance.**

field for LLM responses, where changes are color-highlighted. The latter can be navigated with `Page Up`/`Page Down` or the scrollbar, while typing will modify the query prompt.

## 3 Implementation

LLM-for-X's implementation comprises an operating system (OS)-level background service and a browser extension. The OS service produces its prompt menu UI on demand, inspects selected content inside native apps, retrieves surrounding context for context, inserts responses, and communicates with LLM backend APIs. The browser extension inspects web page contents for selection and inserts responses in web pages. It also has the capability of simulating user interaction with chat-based LLM services as a way to interface with LLM services without requiring a dedicated subscription or paying on a per-request level.

LLM-for-X's prompt menu is the main interactive and user-facing element, configured by our background service. We implemented LLM-for-X to run on Windows 10 and higher. It supports Google Chrome or Microsoft Edge through its extension API. If support for native apps is not needed, LLM-for-X also runs exclusively as a browser extension for use inside browser tabs. This makes it compatible with all platforms that fully support Chrome or Edge and extensions without requiring a native component.

*OS service* Our background app is implemented in C# using .net APIs and registers a system-level keyboard hook to monitor global keyboard shortcuts. When a shortcut is detected, our native app extracts application details and text selection to prepare the prompt.

*Native interface* Our implementation uses the UI Automation API (UIA) on Windows to access UI components and text contents as part of the Microsoft Accessibility API. It additionally obtains the window title and process name of the foreground application to provide additional context for the LLM query.

*Insert LLM responses* Our implementation emulates direct user input into the control to allow responses to flow back via the clipboard. This way, the input operation enters into the foreground application's undo stack and is treated like any other input.

*Web app interface* If the browser is in the foreground, our extension extracts the selected text and surrounding content from the page and later performs text insertion. Our extension is implemented in JavaScript and uses the DOM API for these operations.

*LLM interface.* LLM-for-X currently supports ChatGPT, Mistral, and Gemini. Because these three implement similar chat interfaces, our browser extension injects user input following a query through our popup, obtains the response from the LLM via the chat answer, and transfers it to our prompt dialog. This provides users of our system the direct use of LLM assistance without the need for manual copy & paste or context switching between windows or applications.

Alternatively, LLM-for-X also supports direct interfacing with LLM APIs. This requires a dedicated API subscription and incurs query-specific charges for each use, but it allows for more flexibility over query types and produces faster responses.

## 4 Preliminary evaluation

We evaluated LLM-for-X on authoring, reading, and coding tasks with 14 participants, comparing their performance using either LLM-for-X or ChatGPT 3.5 as the *Interface* . The tasks include summarizing, editing, composing, reading, and coding. Participants completed all three tasks for one condition, took a break, and repeated a secondary instantiation of the tasks in the second condition. Conditions were counterbalanced. An experimenter introduced the study, answered questions, and verified task outcomes.

*Results.* Participants were significantly faster ($Z = -2.18$, $p < 0.05$) during the editing task using LLM-for-X ($M = 31.71$, $SD = 19.04$) than when using ChatGPT ($M = 51.14$, $SD = 32.50$). Participants expressed that they felt more efficient when conducting the tasks with LLM-for-X, because it eliminates the need for context switching. Please see our full analysis for more details [12].

## 5 Conclusion

Through a series of application examples and tasks, this demonstration has showcased the benefit of application-agnostic integration of LLM services. Our shortcut interface allows users to avoid context switches and obtain LLM assistance in place.

## References


[1] Tamara Babaian, Barbara Grosz, and Stuart Shieber. 2002. A Writer's Collaborative Assistant. *International Conference on Intelligent User Interfaces, Proceedings IUI* (06 2002). https://doi.org/10.1145/502716.502722

[2] Carey D Chisholm, Amanda M Dornfeld, David R Nelson, and William H Cordell. 2001. Work interrupted: a comparison of workplace interruptions in emergency departments and primary care offices. *Annals of emergency medicine* 38, 2 (2001), 146–151.

[3] Github. 2024. Github Copilot. https://github.com/features/copilot Accessed: 30-03-2024.

[4] Sophie Leroy, Aaron M Schmidt, and Nora Madjar. 2020. Interruptions and task transitions: Understanding their characteristics, processes, and consequences. *Academy of Management Annals* 14, 2 (2020), 661–694.

[5] André N Meyer, Laura E Barton, Gail C Murphy, Thomas Zimmermann, and Thomas Fritz. 2017. The work life of developers: Activities, switches and perceived productivity. *IEEE Transactions on Software Engineering* 43, 12 (2017), 1178–1193.

[6] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842* (2023).

[7] Bryan Min, Matthew T Beaudouin-Lafon, Sangho Suh, and Haijun Xia. 2023. Demonstration of Masonview: Content-Driven Viewport Management. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (, San Francisco, CA, USA,) *(UIST '23 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 60, 3 pages. https://doi.org/10.1145/3586182.3615827

[8] OpenAI. 2024. Introducing GPTs. https://openai.com/blog/introducing-gpts Accessed: 30-03-2024.

[9] Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*. 117–126.

[10] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2024. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems* 36 (2024).

[11] Shahab Saquib Sohail, Faiza Farhat, Yassine Himeur, Mohammad Nadeem, Dag Øivind Madsen, Yashbir Singh, Shadi Atalla, and Wathiq Mansoor. 2023. Decoding ChatGPT: A taxonomy of existing research, current challenges, and possible future directions. *Journal of King Saud University - Computer and Information Sciences* 35, 8 (2023), 101675. https://doi.org/10.1016/j.jksuci.2023.101675

[12] Lukas Teufelberger, Xintong Liu, Zhipeng Li, Max Moebus, and Christian Holz. 2024. LLM-for-X: Application-agnostic Integration of Large Language Models to Support Personal Writing Workflows. arXiv:2407.21593 [cs.HC] https://arxiv.org/abs/2407.21593

[13] Jiayin Wang, Weizhi Ma, Peijie Sun, Min Zhang, and Jian-Yun Nie. 2024. Understanding User Experience in Large Language Model Interactions. arXiv:2401.08329 [cs.HC]

[14] Writeful. 2024. TexGPT: Harness the power of ChatGPT in Overleaf. https://blog.writefull.com/texgpt-harness-the-power-of-chatgpt-in-overleaf/ Accessed: 30-03-2024.

[15] Julie S Zide, Maura J Mills, Comila Shahani-Denning, and Carolyn Sweetapple. 2017. Work interruptions resiliency: toward an improved understanding of employee efficiency. *Journal of Organizational Effectiveness: People and Performance* 4, 1 (2017), 39–58.